Dymore User's Manual

Domain decomposition approach

Contents

1 Lagrange multipliers

1.1 Classical Lagrange multiplier technique

The continuity of the displacement field across sub-domain boundaries is enforced by imposing linear constraints, the equality of the dofs of corresponding nodes in adjacent sub-domains. Typically, this is achieved by using the classical Lagrange multiplier technique, which is illustrated in a conceptual manner in fig. 1. Let the displacement vectors at two nodes belonging to two adjacent sub-domains be denoted $\underline{u}_1$ and $\underline{u}_2$. The continuity of the displacement field across the interface of the two sub-domains implies $\underline{\mathcal{C}} = \underline{u}_1 - \underline{u}_2 = \underline{0}$, where $\underline{\mathcal{C}}$ is the constraint to be imposed. In the classical Lagrange multiplier technique, the constraint is imposed via the addition of a constraint potential, $V_c = \underline{\lambda}^T \underline{\mathcal{C}}$, where $\underline{\lambda}$ is the array of Lagrange multipliers used to enforce the constraint.

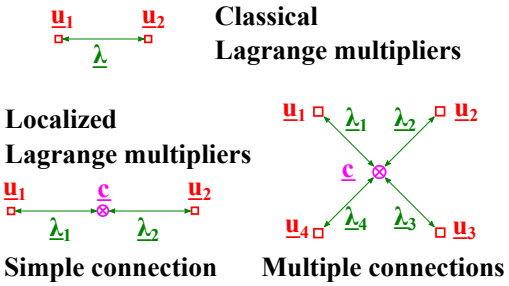


Figure 1: Classical and localized Lagrange multipliers.

1.2 Localized Lagrange multiplier technique

An alternative approach is to define an independent interface node, denoted $\underline{c}$, then impose two kinematic constraints: the displacement components at the boundary nodes in the two sub-domains adjacent to the interface must equal those at the independent interface nodes. For the simple connection illustrated in fig. 1, the two kinematic constraints become $\underline{\mathcal{C}}^{[1]} = \underline{u}_1 - \underline{c} = \underline{0}$ and $\underline{\mathcal{C}}^{[2]} = \underline{u}_2 - \underline{c} = \underline{0}$, and the corresponding constraint potential is $V_c = \underline{\lambda}^{[1]T} \underline{\mathcal{C}}^{[1]} + \underline{\lambda}^{[2]T} \underline{\mathcal{C}}^{[2]}$.

1.3 The localized Lagrange multiplier element

As discussed in section 1.2, the kinematic continuity conditions between sub-domain interfaces is enforced via the localized Lagrange multiplier technique. Let $\underline{u}_b$ and $\underline{c}$ denote the arrays of dofs at a boundary node and at an interface

node, respectively. A typical kinematic constraint is written as $\underline{\mathcal{C}} = \underline{u}_b - \underline{c} = \underline{0}$ and the associated potential is

$$V_c = s\underline{\lambda}^T\underline{\mathcal{C}} + \frac{p}{2}\underline{\mathcal{C}}^T\underline{\mathcal{C}}, \tag{1}$$

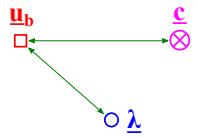
where $\underline{\lambda}$ is the array of Lagrange multipliers used to enforce the constraint, and s the scaling factor for those multipliers. The second term of the potential is a penalty term and p is the penalty coefficient. The potential defined by eq. (1) combines the traditional Lagrange multiplier technique with the penalty method. This combination is known as the augmented Lagrangian formulation and has been studied extensively [1, 2]. It is an effective approach for the enforcement of kinematic constraints in multibody dynamics, as proposed by Bayo *et al.* [3, 4]. Furthermore, scaling of the Lagrange multipliers and the addition of the penalty terms was shown help the solution of differential algebraic equations [5, 6].

A localized Lagrange multiplier element is depicted in fig. 2. It involves three vertices: the first vertex carries the dofs of the boundary node, $\underline{u}_b$, the second those of the interface node, $\underline{c}$, and the last those of the Lagrange multiplier. A variation of the potential defined by eq. (1) is obtained easily,

$$\delta V_c = \delta\underline{u}_b^T\left[s\underline{\lambda} + p\underline{\mathcal{C}}\right] + \delta\underline{\lambda}^T\left[s\underline{\mathcal{C}}\right] + \delta\underline{c}^T\left[-s\underline{\lambda} - p\underline{\mathcal{C}}\right], \tag{2}$$

and gives rise to the following generalized forces of constraint of the localized Lagrange multiplier element,

$$\underline{f} = \begin{Bmatrix} s\underline{\lambda} + p\underline{\mathcal{C}} \\ s\underline{\mathcal{C}} \\ -s\underline{\lambda} - p\underline{\mathcal{C}} \end{Bmatrix}. \tag{3}$$



Localized Lagrange multiplier element

Figure 2: The localized Lagrange multiplier element.

Taking a derivative of these forces of constraint with respect to the dofs of the problem yields the stiffness matrix of the localized Lagrange multiplier element,

$$\underline{\underline{k}} = \begin{bmatrix} p\underline{\underline{I}} & s\underline{\underline{I}} & -p\underline{\underline{I}} \\ s\underline{\underline{I}} & \underline{\underline{0}} & -s\underline{\underline{I}} \\ -p\underline{\underline{I}} & -s\underline{\underline{I}} & p\underline{\underline{I}} \end{bmatrix}, \tag{4}$$

where $\underline{\underline{I}}$ denotes the identity matrix of size $d \times d$. For beam problems, $d = 6$, because each node has six dofs, three displacements and three rotations.

As mentioned in section 1.2, the Lagrange multipliers become local in the proposed formulation, *i.e.*, Lagrange multipliers are associated with a single sub-domain. The potential of kinematic constraint involves two types of dofs, the sub-domain dofs, $\underline{u}_b$ and $\underline{\lambda}$, and the interface dofs, $\underline{c}$. The constraint forces and stiffness matrix are partitioned to reflect this fact

$$\underline{f} = \begin{Bmatrix} \underline{f}_b \\ \underline{f}_c \end{Bmatrix}, \quad \underline{\underline{k}} = \begin{bmatrix} \underline{\underline{k}}_{bb} & \underline{\underline{k}}_{bc} \\ \underline{\underline{k}}_{bc}^T & \underline{\underline{k}}_{cc} \end{bmatrix}. \tag{5}$$

Subscripts $(\cdot)_b$ and $(\cdot)_c$ denote dofs associated with boundary and interface nodes, respectively. Partitioning the constraint forces defined by eq. (3) yields

$$\underline{f}_b = \begin{Bmatrix} s\underline{\lambda} + p\underline{\mathcal{C}} \\ s\underline{\mathcal{C}} \end{Bmatrix}, \quad \underline{f}_c = -\left\{s\underline{\lambda} + p\underline{\mathcal{C}}\right\}. \tag{6}$$

A similar operation for the constraint stiffness matrix leads to

$$\underline{\underline{k}}_{bb} = \begin{bmatrix} p\underline{\underline{I}} & s\underline{\underline{I}} \\ s\underline{\underline{I}} & \underline{\underline{0}} \end{bmatrix}, \quad \underline{\underline{k}}_{cc} = \left[p\underline{\underline{I}}\right], \quad \underline{\underline{k}}_{bc} = \begin{bmatrix} -p\underline{\underline{I}} \\ -s\underline{\underline{I}} \end{bmatrix}. \tag{7}$$

In summary, each kinematic constraint is associated with a localized Lagrange multiplier element, which generates a force vector and a stiffness matrix. Clearly, each kinematic constraint can be viewed as finite element and in the sequel, the terms "kinematic constraint" and "constraint element" will be used interchangeably.

1.4 Assembly procedure for the constraints

In the previous section, the development has focused on a single constraint. To connect the N_s sub-domains, a total of N_c interface nodes will be defined and the following array stores the dofs at all these interface nodes,

$$\underline{c}^T = \left\{\underline{c}_1^T, \underline{c}_2^T, \ldots, \underline{c}_{N_c}^T\right\}. \tag{8}$$

Array $\underline{c}$ is of size n_c. The total potential of all constraints associated with sub-domain i, denoted $V_c^{(i)}$, is found by summing up the potentials of the corresponding constraint,

$$V_c^{(i)} = \sum_{j=1}^{N_b^{(i)}} V_c. \tag{9}$$

Finally, the total potential of all kinematic constraints is

$$V_c = \sum_{i=1}^{N_s} V_c^{(i)}. \tag{10}$$

Each constraint element contributes constraint forces and stiffness matrices defined by eqs. (6) and (7), respectively. Using the standard assembly procedure used in the finite element method [7, 8], the force arrays and stiffness matrices generated by all the constraint elements associated with sub-domain i are assembled in the following global arrays and matrices

$$\underline{F}_b^{(i)} = \sum_{j=1}^{N_b^{(i)}} \underline{\underline{B}}_b^T \underline{f}_b, \quad \underline{\underline{K}}_{bb}^{(i)} = \sum_{j=1}^{N_b^{(i)}} \underline{\underline{B}}_b^T \underline{\underline{k}}_{bb} \underline{\underline{B}}_b, \tag{11}$$

where $\underline{\underline{B}}_b$ is the Boolean matrices used for the assembly process, *i.e.*, $\underline{u}_b = \underline{\underline{B}}_b \underline{u}^{(i)}$. Of course, the assembly procedure can be performed in parallel for all sub-domains. Similarly, the constraint elements contribute force arrays and stiffness matrices to the interface problem,

$$\underline{F}_c^{(i)} = \sum_{j=1}^{N_b^{(i)}} \underline{\underline{B}}_c^T \underline{f}_c, \quad \underline{\underline{K}}_{cc}^{(i)} = \sum_{j=1}^{N_b^{(i)}} \underline{\underline{B}}_c^T \underline{\underline{k}}_{cc} \underline{\underline{B}}_c, \tag{12}$$

where $\underline{\underline{B}}_c$ is the Boolean matrices used for the assembly process, *i.e.*, $\underline{c} = \underline{\underline{B}}_c \underline{c}$. Finally, the constraint coupling stiffness is assembled to find

$$\underline{\underline{K}}_{bc}^{(i)} = \sum_{j=1}^{N_b^{(i)}} \underline{\underline{B}}_b^T \underline{\underline{k}}_{bc} \underline{\underline{B}}_c. \tag{13}$$

2 Solution procedure

In this section, a general solution procedure for block-diagonal systems is presented. For simplicity, the linear system is rewritten as

$$\underline{\underline{A}}\, \underline{x} = \underline{b}, \tag{14}$$

where $\underline{x}$ is the array of unknowns, $\underline{b}$ the known right-hand side, and matrix $\underline{\underline{A}}$ has the following form

$$\underline{\underline{A}} = \begin{bmatrix} \underline{\underline{A}}^{(1)} & \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{A}}_{bc}^{(1)} \\ \underline{\underline{0}} & \underline{\underline{A}}^{(2)} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{A}}_{bc}^{(2)} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{A}}^{(3)} & \cdots & \underline{\underline{0}} & \underline{\underline{A}}_{bc}^{(3)} \\ \vdots & \vdots & \vdots & \ddots & \underline{\underline{0}} & \vdots \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{A}}^{(N_s)} & \underline{\underline{A}}_{bc}^{(N_s)} \\ \underline{\underline{A}}_{bc}^{(1)T} & \underline{\underline{A}}_{bc}^{(2)T} & \underline{\underline{A}}_{bc}^{(3)T} & \cdots & \underline{\underline{A}}_{bc}^{(N_s)T} & \underline{\underline{A}}_{cc} \end{bmatrix}. \tag{15}$$

The procedure described in the previous section leads to system matrices presenting the structure shown in eq. (15).

System (14) will be solved using the classical skyline solver [8], which is based on the factorization of the system matrix as

$$\underline{\underline{A}} = \underline{\underline{L}}\, \underline{\underline{D}}\, \underline{\underline{U}}, \tag{16}$$

where $\underline{\underline{L}}$ and $\underline{\underline{U}}$ are lower and upper triangular matrices, respectively, and $\underline{\underline{D}}$ a diagonal matrix. A fundamental property of the skyline solver is that the skylines of matrices $\underline{\underline{L}}$ and $\underline{\underline{U}}$ are identical to that of matrix $\underline{\underline{A}}$. This implies that matrices $\underline{\underline{L}}$ and $\underline{\underline{U}}$ have the following structure

$$\underline{\underline{L}} = \begin{bmatrix} \underline{\underline{L}}^{(1)} & \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{L}}^{(2)} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{L}}^{(3)} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \vdots & \vdots & \vdots & \ddots & \underline{\underline{0}} & \vdots \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{L}}^{(N_s)} & \underline{\underline{0}} \\ \underline{\underline{M}}^{(1)} & \underline{\underline{M}}^{(2)} & \underline{\underline{M}}^{(3)} & \cdots & \underline{\underline{M}}^{(N_s)} & \underline{\underline{L}}_{cc} \end{bmatrix}, \tag{17}$$

$$\underline{\underline{U}} = \begin{bmatrix} \underline{\underline{U}}^{(1)} & \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{V}}^{(1)} \\ \underline{\underline{0}} & \underline{\underline{U}}^{(2)} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{V}}^{(2)} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{U}}^{(3)} & \cdots & \underline{\underline{0}} & \underline{\underline{V}}^{(3)} \\ \vdots & \vdots & \vdots & \ddots & \underline{\underline{0}} & \vdots \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{U}}^{(N_s)} & \underline{\underline{V}}^{(N_s)} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{U}}_{cc} \end{bmatrix}. \tag{18}$$

Note that matrices $\underline{\underline{M}}^{(i)}$ and $\underline{\underline{V}}^{(i)}$, $i = 1, 2, \ldots, N_s$ are, in general, fully populated matrices. Finally, diagonal matrix $\underline{\underline{D}}$ has the following structure,

$$\underline{\underline{D}} = \begin{bmatrix} \underline{\underline{D}}^{(1)} & \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{D}}^{(2)} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{D}}^{(3)} & \cdots & \underline{\underline{0}} & \underline{\underline{0}} \\ \vdots & \vdots & \vdots & \ddots & \underline{\underline{0}} & \vdots \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{D}}^{(N_s)} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{0}} & \cdots & \underline{\underline{0}} & \underline{\underline{D}}_{cc} \end{bmatrix}. \tag{19}$$

2.1 Details of the factorization procedure

The goal of the factorization procedure is to evaluate all the entries of the lower triangular matrix, $\underline{\underline{L}}$, upper triangular matrix, $\underline{\underline{U}}$, and diagonal matrix, $\underline{\underline{D}}$, defined by eqs. (17), (18), and (19), respectively. The factorization expressed by eq. (16) implies

$$\underline{\underline{A}}^{(i)} = \underline{\underline{L}}^{(i)} \underline{\underline{D}}^{(i)} \underline{\underline{U}}^{(i)}, \quad i = 1, 2, \ldots, N_s. \tag{20}$$

Clearly, the matrices associated with each sub-domain, $\underline{\underline{A}}^{(i)}$, can be factorized independently to find lower and upper triangular matrices $\underline{\underline{L}}^{(i)}$ and $\underline{\underline{U}}^{(i)}$, respectively, and diagonal matrices, $\underline{\underline{D}}^{(i)}$, using the classical skyline solver [7, 8]. Equation (16) also implies

$$\underline{\underline{A}}_{bc}^{(i)} = \underline{\underline{L}}^{(i)} \quad \underline{\underline{D}}^{(i)} \underline{\underline{V}}^{(i)} \quad , \quad i = 1, 2, \ldots, N_s, \tag{21a}$$

$$\underline{\underline{A}}_{bc}^{(i)} = \underline{\underline{U}}^{(i)T} \underline{\underline{D}}^{(i)} \underline{\underline{M}}^{(i)T}, \quad i = 1, 2, \ldots, N_s. \tag{21b}$$

Once matrices $\underline{\underline{L}}^{(i)}$, $\underline{\underline{U}}^{(i)}$, and $\underline{\underline{D}}^{(i)}$ have been obtained, eqs (21a) and (21b) allow the evaluation of matrices $\underline{\underline{V}}^{(i)}$ and $\underline{\underline{M}}^{(i)}$, respectively, using back-substitution. Here again, these operations can be carried out in parallel in each sub-domain.

The last relationship implied by eq. (16) is $\underline{\underline{A}}_{cc} = \sum_{i=1}^{N_s} \underline{\underline{M}}^{(i)} \underline{\underline{D}}^{(i)} \underline{\underline{V}}^{(i)} + \underline{\underline{L}}_{cc} \underline{\underline{D}}_{cc} \underline{\underline{U}}_{cc}$, which lead to the following factorization

$$\bar{\underline{\underline{A}}}_{cc} = \underline{\underline{L}}_{cc} \underline{\underline{D}}_{cc} \underline{\underline{U}}_{cc}, \tag{22}$$

where

$$\bar{\underline{\underline{A}}}_{cc} = \underline{\underline{A}}_{cc} - \sum_{i=1}^{N_s} \underline{\underline{A}}_{cc}^{(i)}, \tag{23}$$

and

$$\underline{\underline{A}}_{cc}^{(i)} = \underline{\underline{M}}^{(i)} \underline{\underline{D}}^{(i)} \underline{\underline{V}}^{(i)}, \quad i = 1, 2, \ldots, N_s. \tag{24}$$

Matrices $\underline{\underline{A}}_{cc}^{(i)}$ can be computed in parallel. All sub-domain contributions are then collected to form matrix $\bar{\underline{\underline{A}}}_{cc}$, the factorization of which yields lower and upper triangular matrices $\underline{\underline{L}}_{cc}$ and $\underline{\underline{U}}_{cc}$, respectively, and diagonal matrix $\underline{\underline{D}}_{cc}$ using the classical skyline solver. This operation completes the factorization of the system matrix according to eq. (16).

2.2 Details of the back-substitution procedure

Once the system matrix has been factorized, the solution is obtained via forward- and back-substitution. The first step of the procedure is to write system (14) as $\underline{\underline{L}}\,\underline{y} = \underline{b}$, where intermediate solution array $\underline{y}$ is defined as $\underline{y} = \underline{\underline{D}}\,\underline{\underline{U}}\,\underline{x}$. This array is partitioned as $\underline{y}^T = \{\underline{y}^{(1)T}, \underline{y}^{(2)T}, \ldots, \underline{y}_c^T\}$, where $\underline{y}^{(i)}$, $i = 1, 2, \ldots, N_s$, are the intermediate solution arrays in each of the sub-domains and $\underline{y}_c$ the corresponding interface quantities.

Given the structure of lower triangular matrix $\underline{\underline{L}}$ expressed by eq. (17), forward-substitution yields the components of intermediate solution array $\underline{y}$ as

$$\underline{\underline{L}}^{(i)}\underline{y}^{(i)} = \underline{b}^{(i)}, \quad i = 1, 2, \ldots, N_s, \tag{25a}$$

$$\underline{\underline{L}}_{cc}\underline{y}_c = \underline{b}_c - \sum_{i=1}^{N_s} \underline{d}^{(i)}, \tag{25b}$$

where

$$\underline{d}^{(i)} = \underline{\underline{M}}^{(i)}\underline{y}^{(i)}, \tag{26}$$

and the right-hand side array was partitioned as $\underline{b}^T = \{\underline{b}^{(1)T}, \underline{b}^{(2)T}, \ldots, \underline{b}_c^T\}$.

Once intermediate solution array $\underline{y}$ is evaluated, intermediate solution array $\underline{z} = \underline{\underline{U}}\,\underline{x}$ is defined, where $\underline{\underline{D}}\,\underline{z} = \underline{y}$. This array is found easily as

$$\underline{\underline{D}}^{(i)}\underline{z}^{(i)} = \underline{y}^{(i)}, \quad i = 1, 2, \ldots, N_s, \tag{27a}$$

$$\underline{\underline{D}}_c\underline{z}_c = \underline{y}_c. \tag{27b}$$

Finally, the solution of the problem is obtained, once from from back-substitution

$$\underline{\underline{U}}_{cc}\underline{x}_c = \underline{z}_c, \tag{28a}$$

$$\underline{\underline{U}}^{(i)}\underline{x}^{(i)} = \underline{z}^{(i)} - \underline{\underline{V}}^{(i)}\underline{x}_c, \quad i = 1, 2, \ldots, N_s, \tag{28b}$$

2.3 Summary of the solution procedure

The input to the factorization procedure are as follows.

1. Stiffness matrices $\underline{\underline{A}}^{(i)}$ and $\underline{\underline{A}}_{bc}^{(i)}$, $i = 1, 2, \ldots, N_s$, for each of the sub-domains. Matrices $\underline{\underline{A}}^{(i)}$ are in compact storage form. According to eq. (7), each constraint applied to sub-domain i generates two non-vanishing entries in $\underline{\underline{A}}_{bc}^{(i)}$; all other columns of this matrix vanish. Hence, these matrices $\underline{\underline{A}}_{bc}^{(i)}$ are not assembled. The columns of matrix $\underline{\underline{A}}_{bc}^{(i)}$ featuring non-vanishing entries are called the "active columns" of that matrix.
2. Interface stiffness matrix $\underline{\underline{A}}_{cc}$. This matrix is in compact storage form.

In summary, the solution procedure can be divided into the five phases detailed below, three of which are parallelized easily.

2.3.1 Phase 1: factorize sub-domain stiffness matrices (parallel)

For each sub-domain, perform the following operations in parallel.

1. Perform the factorization of matrix $\underline{\underline{A}}^{(i)}$ expressed by eq. (20) to find matrices $\underline{\underline{L}}^{(i)}$, $\underline{\underline{D}}^{(i)}$, and $\underline{\underline{U}}^{(i)}$. Matrices $\underline{\underline{L}}^{(i)}$, $\underline{\underline{D}}^{(i)}$, and $\underline{\underline{U}}^{(i)}$ are computed "in place," *i.e.*, they replace matrix $\underline{\underline{A}}^{(i)}$ as the computation proceeds, without additional storage requirement.
2. Evaluate matrix $\underline{\underline{V}}^{(i)}$ with the help of eq. (21a). Each column of this matrix can be found from the corresponding column of matrix $\underline{\underline{A}}_{bc}^{(i)}$ using back-substitution. Clearly, only the active columns of matrix $\underline{\underline{A}}_{bc}^{(i)}$ generate non-vanishing entries in matrix $\underline{\underline{V}}^{(i)}$, *i.e.*, the active columns of matrix $\underline{\underline{V}}^{(i)}$ match those of matrix $\underline{\underline{A}}_{bc}^{(i)}$. Storage must be provided for the active columns of matrix $\underline{\underline{V}}^{(i)}$ only.
3. Evaluate matrix $\underline{\underline{M}}^{(i)}$ with the help of eq. (21b). Each row of this matrix can be found from the corresponding column of matrix $\underline{\underline{A}}_{bc}^{(i)}$ using back-substitution. Storage must be provided for the active rows of matrix $\underline{\underline{M}}^{(i)}$ only.
4. Evaluate matrix $\underline{\underline{A}}_{cc}^{(i)}$ defined by eq. (24). Storage must be provided for this matrix.

In steps 2 and 3, the columns of matrix $\underline{\underline{V}}^{(i)}$ and rows of matrix $\underline{\underline{M}}^{(i)}$, respectively, can all be evaluated independently, providing fine grain parallelelization opportunities.

2.3.2 Phase 2: factorize interface stiffness matrix

To complete the factorization of the system matrix, perform the following operations dealing with the interface stiffness matrix.

1. Evaluate matrix $\bar{\underline{\underline{A}}}_{cc}$ using eq. (23).
2. Factorize matrix $\bar{\underline{\underline{A}}}_{cc}$ according to eq. (22).

2.3.3 Phase 3: forward-substitute in sub-domains (parallel)

Once the system matrix factorization has been completed, the forward-substitution phase can begin.

1. Find the intermediate solution array, $\underline{y}^{(i)}$, in each of the sub-domains via forward-substitution using eq. (25a). Vector $\underline{y}^{(i)}$ is computed "in place," *i.e.*, it replaces vector $\underline{b}^{(i)}$ as the computation proceeds without additional storage requirement.
2. Compute the contribution of the sub-domain to interface forces, $\underline{d}^{(i)}$, using eq. (26). Storage must be provided for vector $\underline{d}^{(i)}$.

2.3.4 Phase 4: solve for interface displacements

The solution of the interface problem proceeds in three steps.

1. Evaluate the right-hand side of eq. (25b) using the intermediate solution arrays, $\underline{y}^{(i)}$, computed in the previous phase. Accumulate sub-domain vectors $\underline{d}^{(i)}$ in place in array $\underline{b}_c$.
2. Find array $\underline{y}_c$ via forward-substitution, then use eq. (27b) to obtain array $\underline{z}_c$.
3. Evaluate interface displacements from eq. (28a) by back-substitution.

2.3.5 Phase 5: solve for sub-domain displacements by back-substitution (parallel)

In the last phase of the solution process, the sub-domain nodal displacements are recovered via back-substitution.

1. Evaluate sub-domain arrays $\underline{z}^{(i)}$ from eq. (27a).
2. Evaluate the right-hand side of eq. (28b).
3. Find sub-domain nodal displacements, $\underline{x}^{(i)}$, via back-substitution using eq. (28b)

References

[1] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*. North-Holland, Amsterdam, the Netherlands, 1983.

[2] P.E. Gill, W. Murray, M.A. Saunders, and M.H. Wright. Sequential quadratic programming methods for nonlinear programming. In E.J. Haug, editor, *Computer-Aided Analysis and Optimization of Mechanical System Dynamics*, pages 679–697. Springer-Verlag, Berlin, Heidelberg, 1984.

[3] E. Bayo, J. García de Jalón, and M.A. Serna. A modified Lagrangian formulation for the dynamic analysis of constrained mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 71:183–195, November 1988.

[4] E. Bayo, J. García de Jalón, A. Avello, and J. Cuadrado. An efficient computational method for real time multibody dynamic simulation in fully Cartesian coordinates. *Computer Methods in Applied Mechanics and Engineering*, 92:377–395, 1991.

[5] C.L. Bottasso, O.A. Bauchau, and A. Cardona. Time-step-size-independent conditioning and sensitivity to perturbations in the numerical solution of index three differential algebraic equations. *SIAM Journal on Scientific Computing*, 29(1):397–414, 2007.

[6] O.A. Bauchau, A. Epple, and C.L. Bottasso. Scaling of constraints and augmented Lagrangian formulations in multibody dynamics simulations. *Journal of Computational and Nonlinear Dynamics*, 4(2):021007 1–9, April 2009.

[7] T.J.R. Hughes. *The Finite Element Method*. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1987.

[8] K.J. Bathe. *Finite Element Procedures*. Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1996.